On Minimizing the Maximum Sensor Movement for Barrier Coverage of a Line Segment

J. Czyzowicz[1], E. Kranakis[2], D. Krizanc[3], I. Lambadaris[4], L. Narayanan[5], J. Opatrny[5], L. Stacho[6], J. Urrutia[7], and M. Yazdani[4]

[1] Département d'informatique, Université du Québec en Outaouais, Gatineau, QC, J8X 3X7, Canada. Supported in part by NSERC grant.
[2] School of Computer Science, Carleton University, Ottawa, ON, K1S 5B6, Canada. Supported in part by NSERC and MITACS grants.
[3] Department of Mathematics and Computer Science, Wesleyan University, Middletown CT 06459, USA.
[4] Department of Systems and Computer Engineering, Carleton University, Ottawa, ON, K1S 5B6, Canada. Supported in part by NSERC and MITACS grants.
[5] Department of Computer Science, Concordia University, Montréal, QC, H3G 1M8, Canada. Supported in part by NSERC grant.
[6] Department of Mathematics, Simon Fraser University, 8888 University Drive, Burnaby, British Columbia, Canada, V5A 1S6. Supported in part by NSERC grant.
[7] Instituto de Matemáticas, Universidad Nacional Autónoma de México, Área de la investigación cientifica, Circuito Exterior, Ciudad Universitaria, Coyoacán 04510, México, D.F. México. Supported in part by CONACYT grant.

Abstract. We consider n mobile sensors located on a line containing a barrier represented by a finite line segment. Sensors form a wireless sensor network and are able to move within the line. An intruder traversing the barrier can be detected only when it is within the sensing range of at least one sensor. The sensor network establishes barrier coverage of the segment if no intruder can penetrate the barrier from any direction in the plane without being detected. Starting from arbitrary initial positions of sensors on the line we are interested in finding final positions of sensors that establish barrier coverage and minimize the maximum distance traversed by any sensor. We distinguish several variants of the problem, based on (a) whether or not the sensors have identical ranges, (b) whether or not complete coverage is possible and (c) in the case when complete coverage is impossible, whether or not the maximal coverage is required to be contiguous. For the case of n sensors with identical range, when complete coverage is impossible, we give linear time optimal algorithms that achieve maximal coverage, both for the contiguous and non-contiguous case. When complete coverage is possible, we give an $O(n^2)$ algorithm for an optimal solution, a linear time approximation scheme with approximation factor 2, and a $(1+\epsilon)$ PTAS. When the sensors have unequal ranges we show that a variation of the problem is NP-complete and identify some instances which can be solved with our algorithms for sensors with unequal ranges.

Key words and phrases: Barrier, Coverage, Detection, Intruder, Line Segment, Optimal Movement, Sensors, NP-complete, PTAS.

1 Introduction

Barrier coverage of a region with wireless sensors is an important application area of sensor networks. Barrier coverage is used to detect intruders attempting to penetrate a protected region. Unlike a complete coverage of a region, it does not necessarily protect the interior points of the region, but rather only its perimeter by detecting intruders that either enter or exit the region. In this respect, therefore, barrier coverage can protect a region with much lower cost in comparison to a complete coverage.

In a general setting of the problem we have a predefined geometric planar region with a well defined boundary and a set of sensors forming a wireless sensor network. The sensors are mobile and they can move with constant and identical speeds in any direction in the plane. Each sensor located at x has a pre-set (determined by the manufacturer) *sensing range* r_x such that any other point p in the plane is within the range of the sensor if and only if its Euclidean distance $d(p, x)$ from x is at most r_x (in the sequel we abbreviate sensing range by *range*). The sensors are initially placed in the plane in arbitrary locations either interior or exterior to the region. Starting from these arbitrary initial positions we are interested in calculating final positions of the sensors where they achieve a barrier coverage of the region, i.e., no part of the boundary is outside of the range of all the sensors, and which minimize the maximum distance traveled by any sensor.

The above optimization problem (referred to as *MinMax* in this paper) arises in a natural way in situations when, due to hostile environment, the sensors cannot be placed initially so that they cover the boundary of the region, but each sensor can be instructed to move into a position in which a barrier coverage of the region is achieved. Since the energy required by a sensor to reach its final positions in the boundary is directly proportional to the distance traveled, minimizing the maximum distance traveled by any sensor minimizes the energy spent by any sensor for reaching its final position. Typically each sensor is battery powered, and thus minimizing the energy spent on moving maximizes the energy available for the subsequent barrier surveillance by the sensor network.

In this paper we restrict our study to the a one dimensional version of the barrier coverage problem. More specifically, a barrier is represented by a finite segment of a line (delimited by its two endpoints), the initial positions of the sensors are arbitrary points on the line, and we consider the problem of optimizing sensor movements within a line, while at the same time achieving a coverage of the barrier. We assume that the intruder moves in a two dimensional plane. Thus an *intruder* may traverse the given barrier from any direction in the plane. As before an intruder can be detected only if it is within the range of at least one sensor of the wireless sensor network and the sensor network establishes barrier coverage if no intruder can penetrate the given line segment in any direction without being detected. Although we consider a simplified version of the general barrier problem, it will become apparent in the sequel that it still contains challenging algorithmic questions and interesting solutions that illustrate the complexity of intrusion detection in this setting.

1.1 Notation and Optimization problems

We now give several preliminary concepts and define more precisely several variants of the barrier coverage problem.

A *barrier* is, without loss of generality, a closed interval $I = [0, L]$ on the real line with pre-defined endpoints 0 and $L > 0$. Consider a set of n points $x_1 \leq x_2 \leq \ldots \leq x_n$ on the real line where x_i represents the initial position of sensor S_i. Let the range of the i-th sensor be denoted by r_i. Then sensor S_i initially covers the closed interval $I(S_i, x_i) = [x_i - r_i, x_i + r_i]$ of length $2r_i$, called the *covering interval* of S_i. It is the set of points of the line (not necessarily in $[0, L]$) which are within the range of the sensor.

We call a *gap* a sub-interval of I none of whose points is within range of any sensor and which cannot be enlarged any further without containing a point within the range of a sensor. Since the ranges of sensors are assumed to be closed intervals, a gap is an open subinterval of $[0, L]$, except when one of the endpoints of the gap is either 0 or L.

The sum of the lengths of all covering intervals is equal to $\sum_{i=1}^{n} 2r_i$ and is denoted by R. Observe that the barrier coverage problem is *feasible* if and only if $R \geq L$, i.e., the sum of the lengths of the covering intervals is at least as large as the length of the interval $[0, L]$.

We investigate how to move the sensors so as to minimize the maximum among the distances traversed by the respective sensors. More formally, if the i-th sensor S_i moves by a distance m_i (a movement to the left will be denoted by $m_i \leq 0$ and movement to the right by $m_i \geq 0$) from its original position x_i, the new position will be $x_i + m_i$ and the new covering interval will be $I(S_i, x_i + m_i)$. Notice that x_i is not assumed to be in the interval $[0, L]$.

If the problem is feasible, i.e., $R \geq L$ then we are interested in studying the following optimization problem.

MinMax optimization problem for $R \geq L$:

$$\text{minimize } \{\max_{1 \leq i \leq n} |m_i|\} \text{ subject to } [0, L] \subseteq \cup_{i=1}^{n} I(S_i, x_i + m_i). \tag{1}$$

When $R < L$ and thus complete coverage of $[0, L]$ is not feasible, we are interested in a *best effort* solution, i.e., an arrangement of sensors that attains the largest possible coverage while at the same time minimizing the maximum movement of sensors. In particular, we consider two variants of the optimization problem previously defined. We call *contiguous* an arrangement of sensors that attains the largest possible coverage as a contiguous sub-interval of $[0, L]$, and *non-contiguous* an arrangement of sensors that attains the largest possible coverage as a collection of possibly disjoint sub-intervals, while at the same time minimizing the maximum movements of the sensors.

Non-contiguous MinMax optimization problem for $R < L$:

$$\text{minimize } \{\max_{1 \leq i \leq n} |m_i|\} \text{ subject to } \cup_{i=1}^{n} I(S_i, x_i + m_i) \subseteq [0, L] \text{ and } |\cup_{i=1}^{n} I(S_i, x_i + m_i)| = R. \tag{2}$$

Contiguous MinMax optimization problem for $R < L$:

$$\text{minimize } \{\max_{1\le i\le n} |m_i|\} \text{ subject to } \cup_{i=1}^{n} I(S_i, x_i + m_i) \subseteq [0, L] \text{ and } \quad (3)$$
$$|\cup_{i=1}^{n} I(S_i, x_i + m_i)| = R \text{ and}$$
$$\cup_{i=1}^{n} I(S_i, x_i + m_i) \text{ is an interval.}$$

We say that a solution of the MinMax problem is *order preserving* if the final positions of the sensors preserve the original ordering of the sensors, in other words, if two sensors on their way to an optimal location never need to cross paths. The existence of order preserving solutions will be useful in finding efficient algorithms for the Min-Max optimization problems for sensors with identical ranges and also for some more general instances of sensors with non-identical ranges specified below.

Lemma 1 (Order Preservation). *Let $S_1, S_2, \ldots, S_n$ be sensors with ranges $r_1, r_2, \ldots, r_n$ in initial positions $x_1 \le x_2 \le \cdots \le x_n$. If there are no two sensors S_i and S_j, $1 \le i \ne j \le n$ such that $x_j - r_j < x_i + r_i$ and $x_j + r_j > x_i + r_i$ then there is an order-preserving optimal solution of any of the three versions of the MinMax optimization problem.*

Proof. (Outline) Consider a solution of a MinMax problem in which two consecutive sensors are out of order, i.e., there are sensors S_i and S_j, $i < j$ and the final position of sensor S_j precedes the final position of S_i. It can be easily seen that we can reverse the order of these two sensors so that they cover the same area, or same size area in case $R < L$, without increasing the value of the maximal move in the solution. Thus by a sequence of switches we can obtain an optimal solution that preserves the original order of sensors.

It is easy to see that an order preserving solution does not need to exist when the conditions of Lemma 1 are not satisfied. The hypothesis of Lemma 1 is clearly satisfied when the covering intervals of the sensors form a *proper interval graph*, i.e., an interval graph that has an intersection model in which no interval properly contains another (see [6]), and obviously in the case of sensors with identical ranges. We also note that the special case of the order preservation lemma for sensors with identical ranges was already stated in [2].

1.2 Related work

In the area of sensor networks, several recent papers considered the problem of deployment of mobile sensors for coverage of a region, see for example [10], [11], and [12]. Unlike the problem considered in this paper, they aim to provide coverage of the inside of a two-dimensional region, and they do not consider the optimization problems stated above. The problem studied in our paper addresses the problem of ensuring efficient border surveillance of a region and intruder detection using a wireless sensor system without covering the inside of the region.

In [9] efficient algorithms are proposed to determine, after sensor deployment, whether a region is barrier covered. It also establishes optimal deployment patterns to achieve barrier coverage when deploying sensors deterministically. In addition, they consider barrier coverage with high probability when sensors are deployed randomly. The problem of local barrier coverage is introduced in [4]. It shows that it is possible for individual sensors to locally determine the existence of local barrier coverage, even when the region of deployment is arbitrarily curved. Techniques for deriving density estimates for achieving barrier coverage and connectivity in thin strips are studied in [1], where sensors are deployed as a barrier to detect moving objects. In all these instances the problem studied concerns *static* optimal sensor deployment patterns and there is no concept of movement of the sensors.

Related to our study is the *Earth Movers Problem* (or EMP) (see [5], [3], [8]). In an ESP problem we have a set of suppliers A which are to move *earth* to the set B of receivers (or holes). The Earth Movers Distance (EMD) of A to B measures the minimum amount of work needed to fill the holes with earth and the aforementioned papers look at optimizations for specific types of motions of the point sets. Results of these papers are not applicable to the optimal movement barrier coverage problem considered in our paper and, despite some similarities, EMP differs from our problem since in barrier coverage there are no fixed destinations, and global coverage should be accomplished with minimizing the maximal distance to final positions.

The most directly related research is the work in [2] where a simpler problem was introduced and studied. Their optimization problem is similar but it differs from our model in that they do not specify the sensor ranges to be employed; unlike in our paper they seek algorithms to move the sensors to equidistant locations on the barrier so as to optimize the efficiency of the barrier coverage regardless of the initial coverage of the sensors. This is generally simpler to accomplish than the problem proposed here. In our work the algorithm is sensitive to the predefined sensor ranges (which are given as input to the problem) thus accomplishing the same barrier coverage task with less movement than may be necessary in [2].

1.3 Results and outline of the paper

In this paper we give several efficient algorithms to solve the MinMax optimization problems stated above. We distinguish several interesting variants of the barrier coverage problem, based on (a) whether or not the sensors have identical ranges, (b) whether or not complete coverage is possible and (c) in the case when complete coverage is impossible, whether or not the maximal coverage is required to be contiguous.

All our algorithms are centralized: they are given initial positions of sensors and they calculate optimized final positions.

Table 1 summarizes results of the paper for the case of sensors with identical range, say r. In the table n is the number of sensors, $R = 2nr$ is the sum of lengths of covering intervals of all the sensors, and L the length of the barrier to

be covered by the sensors. Clearly, the case of sensors with identical range would be very common in practice, since in many sensor networks all sensors are made by the same manufacturer.

coverage		contiguous	non-contiguous
$R < L$		$O(n)$	$O(n)$
$R = L$		$O(n)$	n.a.
$R > L$	optimal	$O(n^2)$	n.a.
	2-approximation	$O(n)$	n.a.
	$1+\epsilon$ approximation	$O\left(n \log\left(\frac{\log(C/g)}{\log(1+\epsilon)}\right)\right)$	n.a.

Table 1. Results for the MinMax problem assuming the n sensors have identical ranges. L is the length of the barrier and R the sum of length of covering intervals, and C, g are both linear functions of the initial sensor positions and the length of the line segment to be covered.

When the sensors have unequal ranges, it is an open problem whether or not the MinMax optimization problem is NP-complete in general. However a variation of the MinMax problem whereby one of the sensors is assigned a fixed position is shown to be NP-complete. We also identify some instances of sensors that have unequal ranges which can be solved with our algorithms for sensors with equal ranges.

The paper is organized as follows. Sections 2, 3, 4, and 5 deal with sensors having identical ranges. Section 2 and 3 provide algorithms for the contiguous coverage, Section 5 for non-contiguous coverage and Section 4 approximation algorithms. Section 6 presents two results for sensors with unequal ranges. First, if the sensor coverage intervals satisfy the condition of Lemma 1 then the algorithmic solutions to the MinMax problems previously stated are still valid. Second, we show that a variation of the MinMax problem is NP-complete. The paper concludes with several proposals for extensions as well as related open problems. Due to the page limit some proofs are abridged or omitted.

2 Contiguous MinMax optimization problem for $R < L$

To solve the contiguous MinMax optimization problem for $R < L$, we proceed as follows: we first solve the problem of covering an interval of size R on the infinite line while minimizing the maximal movement of any sensor, and then we show how to modify our solution so that the interval covered is a subinterval of $[0, L]$.

Lemma 2. *Let $S_1, S_2, \ldots, S_n$ be n sensors with identical range r located on a line in initial positions $x_1 \leq x_2 \leq \ldots \leq x_n$. There is an $O(n)$ time algorithm that calculates the movements of sensors on the line so that the sensors cover a segment of the line of size $2rn$ and the maximal movement of any sensor is minimized.*

When sensors are in the positions determined by the algorithm of the previous theorem they give a maximal contiguous coverage of a segment of the line with a minimized maximal shift, but they do not necessarily cover the segment $[0, L]$. However, when $R < L$ we can easily modify the solution above to achieve a maximal contiguous coverage of $[0, L]$.

Theorem 1. *Let $S_1, S_2, \ldots, S_n$ be n sensors with identical range r located on a line in initial positions $x_1 < x_2 < \ldots < x_n$ and $R < L$. There is an $O(n)$ time algorithm that solves the MinMax optimization problem of covering a line segment of size R of $[0, L]$ so that the maximal value of the shift of any sensor is minimized.*

3 MinMax optimization problem for $R \geq L$

Since the solution to the MinMax problem for $R = L$ consists simply of moving S_i to position $(2i-1)r$, we discuss below only the case $R > L$. We begin with a lemma that provides sufficient conditions for optimality. We say that a set of sensors $S_i, S_{i+1}, \ldots, S_j$ with $j \geq i$ are *in attached position* if any two consecutive sensors in the sequence are exactly $2r$ distance apart.

Lemma 3. (Sufficient condition for optimality) *Let S be an order preserving solution to the MinMax problem by sensors $S_1, S_2, \ldots, S_n$ of identical range with $R > L$. Let x be the largest shift value of any sensor in this solution. If the solution satisfies one of the following conditions, then x is the largest shift value in any optimal solution to the MinMax problem of this instance:*

(a) $x = 0$.
(b) There exists a pair of sensors S_i and S_j, $i < j$ such that the right shift of S_i is equal to x, the left shift of S_j is equal to x, and sensor $S_i, S_{i+1}, \ldots S_j$ are in attached positions.
(c) There is a sensor S_i whose left shift value is equal to x and all sensors preceding S_i are in attached positions up to position 0.
(d) There is a sensor S_i whose right shift value is equal to x and all sensors following S_i are in attached positions up to position L.

We now give an optimal algorithm to solve the MinMax problem. The key idea is to cover the gaps one by one from left to right, while balancing the cost of covering it from the left versus covering it from the right as much as possible.

Min-max Algorithm
Input: L, n, and $x_1 < x_2 < \ldots < x_n$, the initial positions of sensors $S_1, S_2, \ldots, S_n$ with $R > L$.

Let $rmax$ and $lmax$ be the current maximum right and left shift respectively experienced by any sensor and let $x = max(lmax, rmax)$. Initially $rmax = lmax = 0$. We specify how to cover gap g_i while maintaining as an invariant the disjunction of the conditions (a),(b),(c), or (d) of Lemma 3.

Obviously the invariant (in particular, condition (a)) holds at the start of the algorithm. We only need to consider the situation when there is a gap in the given interval $[0, L]$ that is not covered by any sensor. If the given instance contains a gap starting at 0 and there is no sensor to the left of 0 then this

first gap must be covered by shifting to the left sensors of the leftmost group of sensors, (i.e., sensors between this gap and the second gap) in attached position as needed. Then the condition (c) is satisfied at 0 with x equal to the size of the first gap.

Assume one of the conditions (a), (b), or (c) holds just before we cover gap g_i. (We will show that condition (d) can be satisfied only after the last gap.) If $x = 0$ we set inv to be the leftmost sensor. Otherwise, if condition (b) holds then let inv be the rightmost node such that its left shift equals x and such that it is preceded by a node whose left shift equals x, and all intermediate nodes are in attached position. For brevity, we will say that the condition (b) holds at node inv. If condition (c) holds then inv is the rightmost node such that its left shift equals x and all nodes preceding it are in attached position. For brevity, we will say that the condition (c) holds at node inv.

We define $lsurplus(g_i)$ to be the *surplus sensor range* starting at inv up to the node $lnode(g_i)$ and whose right shift would not increase the right shift of sensors past x. Thus $lsurplus(g_i) \leq x$.

Step 1: Move $lnode(g_i)$ to the right by an amount $m = min(lsurplus(g_i), b_i - a_i)$. The nodes to the left of $lnode(g_i)$ follow in attached position as needed.

Observe that the left shift of all nodes in this move cannot increase (and may decrease), and the right shift of any node is at most x, and the invariant (condition (b) or (c)) still holds at node inv. If $m = b_i - a_i$, we have covered gap g_i, and we can move on to the next iteration, that is, to cover gap g_{i+1}.

If instead $m < b_i - a_i$, the gap g_i is not covered completely, and two possibilities exist.

$m = x$**:** In this case, the right shift of node $lnode(g_i)$ is equal to x. Thus, if moved further to the right, the value of its right shift will exceed x.

$m < x$**:** In this case, the right shift of node $lnode(g_i)$ is less than x, but all nodes to the left of it are in attached positions up to a node $prev(g_i)$ whose right shift is equal to x.

Step 2: If $lnode(g_i)$ is the rightmost sensor move $lnode(g_i)$ to the right until its range reaches L. The nodes to the left of $lnode(g_i)$ follow in attached position as needed. Clearly, condition (d) holds after this operation at position L. If $lnode(g_i)$ is not the rightmost sensor, we cover the remainder of gap g_i using sensors to the right of the gap g_i as much as possible using left shifts up to x. We move $rnode(g_i)$ to the left by an amount $m' = min(x, b_i - (a_i + m)$, with the nodes between $rnode(g_i)$ and $lnode(g_{i+1})$ (or the leftmost sensor if g_i is the last gap) following in attached position as needed.

Once again, this move preserves the invariant at node inv; no node increases its right shift, and the left shift of nodes does not exceed x. If $m' = b_i - (a_i + m)$, we have covered the gap g_i and we can move on to the next iteration, that is, to cover gap g_{i+1}. Otherwise, $m' = x$ and the gap g_i is not completely covered. Further, the remainder of this gap cannot be covered from the right or left without increasing the shift of some node to more than the value x.

Step 3: Let y be the midpoint of the gap that remains. We now move $lnode(g_i)$ to $m = min(y, 2r(lnode(g_i) - r)$, with nodes to the left of it following in attached

position as needed. At the same time, we also move $rnode(g_i)$ to the same position $m + 2r$, with the nodes between $rnode(g_i)$ and $lnode(g_{i+1}$ following in attached position as needed.

Clearly, at this point the gap g_i is covered completely. If $m = y$, then observe that $lmax = rmax$. Indeed in Case 1, the invariant (b) now holds at node $rnode(g_i)$, since $rnode(g_i)$ and $lnode(g_i)$ both now have the maximum value of shift, and trivially all nodes between $rnode(g_i)$ and $lnode(g_i)$ are in attached position. In Case 2, recall that all nodes between $prev(g_i)$ and $lnode(g_i)$ were already in attached position; Step 3 ensures that the maximum right shift and left shift are experienced by $prev(g_i)$ and $rnode(g_i)$ with all nodes in between in attached position. Therefore, condition (b) now holds either at node $rnode(g_i)$ or a node between $rnode(g_i)$ and $lnode(g_{i+1}$ whose left shift is equal to that of $rnode(g_i)$.

If instead $m < y$, it means that the sensor nodes to the left of this point do not have sufficient range to cover the interval $[0, y]$. Indeed, Step 3 results in moving the sensor nodes up to $rnode(g_i)$ in attached position starting at 0, thereby making $lmax > rmax$ but ensuring that invariant (c) holds at node $rnode(g_i)$.

Observe that moving the nodes from the right of the gap g_i may result in creating a new gap, one with no nodes to the right of it. However this can only happen once, and only new gap may be introduced in the course of the algorithm.

Since condition d can hold only after the last gap is covered, the preceding arguments show that the invariant is always maintained by the algorithm.

Theorem 2. *Let $S_1, S_2, \ldots, S_n$ be sensors of identical range in initial positions $x_1 \leq x_2 \leq \ldots \leq x_n$ with $R > L$. The Min-max algorithm specified above solves the MinMax problem of covering of the line segment $[0, L]$ so that the maximal value of the shift of any sensor is minimized in time $O(n^2)$.*

4 Approximation Schemes for the MinMax problem for $R > L$

In this section we present two approximation algorithms. The first is an $O(n)$ (linear time) algorithm which is 2 times optimal and the second a $1 + \epsilon$ approximation scheme with running time $O\left(n \log\left(\frac{\log(C/g)}{\log(1+\epsilon)}\right)\right)$, where $\epsilon > 0$, g is the length of the largest gap in the original configuration of n sensors on a line segment of length L, and both C, g are linearly dependent on L.

4.1 Linear time 2-approximation scheme

Assume the initial positions of the n sensors create m gaps in the interval $[0, L]$ and that for each sensor i we have $r \leq x_i \leq L - r$. Since $R > L$, the entire interval can be covered by the n sensors. We need to find a way to move a subset of sensors so that the entire interval is covered, and the maximum movement over all sensors is minimized. We now show a 2-approximation algorithm for this problem.

Theorem 3. *Let $S_1, S_2, \ldots, S_n$ be sensors of identical range in initial positions $x_1 \leq x_2 \leq \ldots \leq x_n$ with $R > L$. There is a linear time 2-optimal approximation scheme for the MinMax optimization problem.*

4.2 $1 + \epsilon$ approximation scheme

Theorem 4. *Let $S_1, S_2, \ldots, S_n$ be sensors of identical range in initial positions $x_1 \leq x_2 \leq \ldots \leq x_n$ with $R > L$. There is a polynomial time approximation scheme which for any $\epsilon > 0$ gives an arrangement of sensors such that the maximum movement achieved is $(1 + \epsilon)$ of the optimal solution in time*

$$O\left(n \log\left(\frac{\log(C/g)}{\log(1+\epsilon)}\right)\right),$$

where $\epsilon > 0$, and C and g are both linear functions of L and the initial sensor positions.

Proof. Let g be half of the length of the largest gap in the original configuration of n sensors divided by 2 on a line segment of length L. It is clear that the value of the MinMax is bounded from above by $C = \max\{L, |x_1|, |L - x_n|$, and from below by g, since a sensor must move at least a distance equal to half the largest gap. Now we specify the two steps used in the algorithm.

Step 1. For a given value of M, where $0 < M \leq C$, check if you can do coverage making movements that do not exceed the value M. Essentially, this involves using the greedy algorithm to move sensors one by one to cover each gap: first move sensors from the left of the gap and if you run out use sensors from the right of the gap. There are two cases. Either it is possible to do coverage using at most M in which case you set $M \leftarrow M/2$ and iterate or else it is not possible to do coverage using at most M in which case you set $M \leftarrow 2M$ and iterate.

Step 2. Now we can give the $1+\epsilon$ approximation. Given $\epsilon > 0$ do a binary search to test each of the potential values $g(1+\epsilon), g(1+\epsilon)^2, \ldots, g(1+\epsilon)^k, \ldots$ for M and find the smallest value of k such that $g(1+\epsilon)^k$ is sufficient in Step 1 but $g(1+\epsilon)^{k-1}$ is not. Observe that if M is the optimal value then $g(1+\epsilon)^{k-1} < M \leq g(1+\epsilon)^k$. It follows that

$$k \leq \frac{\log(M/g)}{\log(1+\epsilon)} \leq \frac{\log(C/g)}{\log(1+\epsilon)}.$$

Moreover, it is easy to see that the resulting approximation factor is $1 + \epsilon$ and the running time as desired, which proves the theorem.

5 Algorithm for the Non-contiguous MinMax problem

As remarked on earlier, this problem only applies for the case $R < L$. We first show how to solve the MinMax problem on the infinite line. Define $le(I)$ and $re(I)$ to be the left endpoint and right endpoint (respectively) of an interval I

on the infinite line. As before $I(S_i, x_i) = [x_1 - r, x_1 + r]$ is the covering interval of sensor S_i. We assume that the initial sensor positions x_i are sorted. The solution to the problem is a set of final positions y_i for the sensors. In the non-contiguous case, the sensor ranges corresponding to the final positions coalesce into a *set* of intervals, rather than a single interval as in the contiguous case. In fact, the final positions can be represented as a set of disjoint intervals $L = \{L_1, \ldots L_k\}$. where each L_t corresponds to a set of sensors, the length of each L_t is a multiple of $2r$ and $\Sigma_{t=1}^{k} |L_t| = 2nr$. As a consequence of the order preservation lemma, the final positions of the sensors can then be derived from the set L quite easily. In particular, to derive the position of sensor S_i, let m be such that $\sum_{t=1}^{m-1} |L_t| < 2ri \leq \sum_{t=1}^{m} |L_t|$. Then the final position y_i of sensor S_i will be $le(L_m) + (i - u - 1)2r + r$ where $u = \sum_{t=1}^{m-1} |L_t|/(2r)$. We thus can prove the following theorem.

Theorem 5. *Let $S_1, S_2, \ldots, S_n$ be sensors of identical range in initial positions $x_1 \leq x_2 \leq \ldots \leq x_n$ with $R < L$. There is a linear time algorithm for the non-contiguous MinMax optimization problem.*

6 Unequal Sensor Ranges

In this section we look at the barrier coverage problem for sensors with non-identical ranges.

6.1 Algorithms for sensors with non-identical ranges

The results of the previous sections can be easily generalized to some types of instances of sensors with unequal sensor ranges. The key part of the algorithms in Sections 2, 3 and 5 is the existence of a solution of the MinMax problem in which the sensors preserve the initial order. Notice that in the algorithms for the MinMax problem for $R < L$ and for $R > L$, we only used the properties of the order preservation lemma in the solution, and, therefore, these algorithms also can be used to solve the MinMax problems for $R < L$ and $R > L$ for any instance of sensors with unequal ranges satisfying the order preservation lemma. The appropriate setting for generalizing these results is for the covering intervals of the sensors to satisfy the condition of Lemma 1. In particular, we mention without proof the following theorem which generalizes the results in Sections 2, 3 and 5.

Theorem 6. *Let $S_1, S_2, \ldots, S_n$ be sensors of ranges $r_1, r_2, \ldots, r_n$ in initial positions $x_1 \leq x_2 \leq \ldots \leq x_n$ such that the covering intervals $I(S_1, x_1), I(S_2, x_2), \ldots, I(S_n, x_n)$ satisfy the condition of Lemma 1.*
If $R < L$ then the algorithm from Theorem 1 solves the contiguous MinMax optimization problem in time $O(n)$.
If $R \geq L$ then the algorithm from Theorem 2 solves the MinMax optimization problem in time $O(n^2)$.

6.2 NP completeness

We now prove an NP-completeness result for sensors with non-identical ranges. We show that a variation of the MinMax problem, where one sensor is assigned a predetermined position, is NP-complete (the sensor with predetermined position could correspond to a sink sensor in practical applications).

Theorem 7 (Case $R > L$). *Consider $n > 1$ sensors $S_1, S_2, \ldots, S_n$ having ranges $r_1, r_2, \ldots, r_n$, respectively, and located in initial positions $x_1 \leq x_2 \leq \ldots \leq x_n$ on a line segment $[0, L]$, Assume further that the final position of sensor S_1 must be equal to a given value z of the segment, and $\sum_{i=1}^{n} 2r_i > L$. The problem of determining for a given k whether there exist final positions of sensors on the line so that the sensors cover the segment $[0, L]$ and the maximum movement of any sensor is at most k is NP-hard.*

Proof. We prove it by reducing the *Partition problem* (see [7][page 47]) into a problem of covering a line segment with sensors such that one sensor must be in a pre-determined final position and the maximum movement of sensors is bounded by a given value. The Partition problem is defined as follows: given a sequence of integers $a_1 \geq a_2 \geq \ldots \geq a_m$, determine whether there exists a set of indices J such that $\sum_{i \in J} a_i = \frac{1}{2} \sum_{i=1}^{m} a_i$.

Let $C = (\sum_{i=1}^{n} a_i)/2$ and consider the barrier coverage problem of segment $I = [0, L]$ where $L = 1 + 4C$, one sensor S_1 of range $1/2$ must be in a pre-determined final position equal to $L/2$, there is one sensor S_{i+1} of range $a_i/2$ for every $1 \leq i \leq m$, located initially in the middle of the line segment. Also, there are two additional sensors S_{m+2}, S_{m+3} of range $(C + a_1)/2$, initially located just outside I so that they cover only points 0 and L of the interval, respectively (see Figure 1). Now if there is a set of indices J such that $\sum_{i \in J} a_i = (\sum_{i=1}^{n} a_i)/2$,

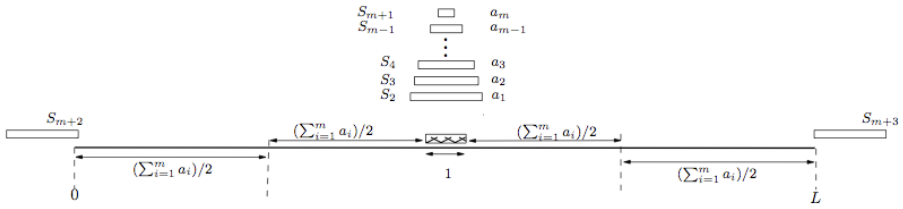

Fig. 1. Arrangement of sensors for proving the NP-completeness of a slightly restricted variation of the MinMax problem when one sensor cannot be moved.

there is a solution to the barrier coverage problem such that for any $i \in J$ the sensor S_{i+1} is moved to the left of the interval covered by S_1 and for any $i \notin J$ the sensor S_{i+1} is moved to the right of the interval covered by S_1. Thus, this way we can cover regions of size C to the left and right of the covering interval of S_1 with all shifts being at most of size C. The two regions at the left and right end of $[0, L]$ can be covered by S_{m+2} and S_{m+3} with shifts being at most C.

If such a partition does not exist, then any distribution of sensors with ranges $a_1/2, a_2/2, \ldots, a_n/2$ to the left and right of S_1 in the predetermined position covers a region of size less that C on one side of the region covered by S_1. Therefore, we have to move one of the sensors at one end of the interval more than C to get a solution. It is easy to observe that ranges of sensors S_{m+2} and S_{m+3} are large enough to have a solution.

Thus if there is an algorithm that can determine if there are movements of sensors on the line so that one sensor is in a predetermined position, the sensors cover the segment $[0, L]$, and the maximum movement of any sensors is at most C, we can determine whether the partition problem has a solution. Clearly, the transformation from the partition problem to the sensor movement problem is polynomial.

Notice that this result also implies NP-completeness of a generalization of the MinMax problem in which the barrier consists of more than one segment.

7 Conclusion and open problems

We have studied the barrier coverage problem for a wireless sensor network when the barrier is a finite line segment. In addition to investigating trade-offs and algorithms with improved running time, the following problems are worth investigating and exploring further. First of all for the case of a line segment, considering (a) the problem of barrier k coverage, whereby each intruder should be detected by at least k different sensors, for some fixed $k \geq 1$, (b) the possibility that there are specified zones which do not need (or are not allowed) to be covered by sensors. Another class of problems concerns extensions to higher dimensions. Note that the two dimensional version of the problem is wide open. More specifically it is worth considering the class of problems for other more general geometric barriers, e.g., circular barriers, convex barriers and more generally boundaries of simplex polygons. It is also worth considering other types of sensor movements, e.g., the movement of the sensors towards the globally optimal position on the circular barrier may proceed through the interior of the circle as opposed to only moving on the perimeter. Finally it is worth exploring the case where the relative sensor ranges are bounded, e.g., $b \leq \frac{r_i}{r(S_j)} \leq B$, for all sensors S_i, S_j, for some constants b, B independent of the number of sensors n. The complexity of the general MinMax problem should be answered as well.

Bibliography

[1] P. Balister, B. Bollobas, A. Sarkar, and S. Kumar. Reliable density estimates for coverage and connectivity in thin strips of finite length. *Proceedings of the 13th annual ACM international conference on Mobile computing and networking*, pages 75–86, 2007.

[2] B. Bhattacharya, M. Burmester, Y. Hu, E. Kranakis, Q. Shi, and A. Wiese. Optimal Movement of Mobile Sensors for Barrier Coverage of a Planar Region. *Theoretical Computer Science, to appear. Also in proceedings of 2nd Annual International Conference on Combinatorial Optimization and Applications (COCOA'08) held August 21-24, 2008, in St. John's, Newfoundland, Canada. LNCS, Vol. 5165*, pages 103–115, 2008.

[3] S. Cabello, P. Giannopoulos, C. Knauer, and G. Rote. Matching point sets with respect to the Earth Mover's Distance. *Computational Geometry: Theory and Applications*, 39(2):118–133, 2008.

[4] A. Chen, S. Kumar, and T.H. Lai. Designing localized algorithms for barrier coverage. *Proceedings of the 13th annual ACM international conference on Mobile computing and networking*, pages 63–74, 2007.

[5] S. Cohen. *Finding Color and Shape Patterns in Images.* PhD Thesis, Stanford University, Dept. of Computer Science, 1999.

[6] P.C. Fishburn. *Interval Orders and Interval Graphs.* Wiley, New York, NY, 1985.

[7] M.R. Garey and D.S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-completeness.* WH Freeman, San Francisco, 1979.

[8] O. Klein and R.C. Veltkamp. Approximation Algorithms for Computing the Earth Mover's Distance Under Transformations. *Lecture Notes in Computer Science*, 3827:1019, 2005.

[9] S. Kumar, T.H. Lai, and A. Arora. Barrier coverage with wireless sensors. *Wireless Networks*, 13(6):817–834, 2007.

[10] X. Li, H. Frey, N. Santoro, and I. Stojmenovic. Localized sensor self-deployment with coverage guarantee. *ACM SIGMOBILE Mobile Computing and Communications Review*, 12(2):50–52, 2008.

[11] Shuhui Yang, Minglu Li, and Jie Wu. Scan-based movement-assisted sensor deployment methods in wireless sensor networks. *IEEE Trans. Parallel Distrib. Syst.*, 18(8):1108–1121, 2007.

[12] Y. Zou and K. Chakrabarty. A distributed coverage- and connectivity-centric technique for selecting active nodes in wireless sensor networks. *IEEE Trans. Comput.*, 54(8):978–991, 2005.

Appendix

Proof. (of Lemma 1) Consider a solution of the MinMax problem in which two consecutive sensors are out of order, i.e., there are sensors S_i and S_j, $i < j$ and the final position of sensor S_j precedes the final position of S_i. It can be easily seen that the reversing of the order of these two sensors cannot increase the value of the maximal move in the solution, and thus by a sequence of switches we can obtain an optimal solution that preserves the original order of sensors.

Proof. (of Lemma 2) Let $y_1, y_2, \ldots, y_n$ be positions on the line such that when sensor $S_1, S_2, \ldots, S_n$ move to position $y_1, y_2, \ldots, y_n$ respectively, the sensors cover a contiguous segment of the line of size $2rn$ and the value $\max\{|x_i - y_i| : 1 \leq i \leq n\}$ is minimal among all such possible assignment of values $y_1, y_2, \ldots, y_n$. Notice first that according to Lemma 1, there is an optimal solution to the problem of minimizing the maximal movement of any sensor such that $y_1 < y_2 < \ldots < y_n$. Furthermore, since the sensors cover a contiguous segment of the line, we have $y_i = y_1 + 2(i-1)r$ for $2 \leq i \leq n$ in an optimal solution. Our algorithm determines a solution of this type.

Consider the possibility that the sensors $S_1, S_2, \ldots, S_n$ have moved to positions $y_1 = 0, y_2 = 2r, \ldots, y_n = 2(n-1)r$, respectively on the line, i.e., the sensor S_1 moved to location 0 and the other sensors moved to the subsequent location to the right of it to achieve maximal contiguous coverage. Then the values $-x_1, 2r - x_2, \ldots, 2(n-1)r - x_n$ give the displacements of the sensors, the negative values indicating a shift to the left, positive values indicating a shift to the right on the line and the absolute value of the smallest negative value gives the maximal movement of any sensor to the left, largest positive value gives the maximal movement of any sensor to the right. If the position 0 of sensor S_1 is increased/decreased by c and we shift the positions of other sensors in the same direction by c so that we maintain the maximal contiguous coverage of a line segment then values of all left shifts of sensors are decreased/increased by c, and values of all the right shifts of sensors are increased/decreased by c. Let z_1 be the maximal value and z_2 be the smallest value in the list $x_1, x_2 - 2r, \ldots, x_n - 2(n-1)r$. Thus when we select $c = -(z_1 + z_2)/2$, we achieve a balance between the maximal shift to the right and maximal shift to the left of sensors. Thus any other shift cannot create a smaller maximal shift to the left and to the right. Therefore, $y_i = 2r(i-1) - (z_1 + z_2)/2$ is the position of S_i that minimizes the maximal shift. Clearly, the n values $y_i = 2r(i-1) - (m_1 + m_2)/2$, $1 \leq i \leq n$ can be calculated in $O(n)$ time.

Proof. (of Theorem 1) We calculate using the linear time algorithm given in the proof of Lemma 2 the maximal contiguous coverage of a segment of a line, i.e., of size R, which minimizes the maximal move of any sensor. Let c be the position of S_1 in the solution. If $r \leq c \leq r + L - R$, then the sensors already cover a segment of size R of the interval $[0, L]$ and we are done. Otherwise, we consider two cases:

If $c < r$ then the optimal solution of the previous theorem covers a portion of the line to the left of 0. Thus we shift the positions of the sensors to the right

by assigning to S_i position $y_i = r + 2r(i-1)$. Clearly, this shift will increase the maximal right shift of sensors and decrease the left shift of sensors, but no other solution can have a smaller right shift.

If $r + L - R < c$ then the optimal solution of the previous theorem covers a portion of the line to the right of L. Thus we shift the positions of the sensors to the left by assigning to S_i position $y_i = L - r - 2r(n-i)$. Clearly, this shift will increase the maximal left shift and decrease the right shift of sensors, but no other solution can have smaller left shift.

Since the modification of the solution obtained by the $O(n)$ algorithm of Lemma 2 requires only $O(n)$ additional operations, the entire algorithm takes linear time.

Proof. (of Lemma 3) Let S be a solution of an instance of the MinMax problem with $R > L$ satisfying the condition of the lemma. Let S' be an optimal solution to the same problem with maximum shift less than x. According to the order preservation lemma, we only need to consider S' that preserves the original order of sensors. We have four cases to consider:
Condition (a) is satisfied by S: Obviously no solution could have a maximum shift less than 0.
Condition (b) is satisfied by S: Let p be the position of sensor S_i in solution S. Clearly S' must place S_i in position $p - \epsilon$ for some $\epsilon \geq 0$. If $\epsilon > 0$ then the positions of sensors $S_{i+1}, S_{i+2}, \ldots S_j$ in solution S' must be also shifted to the left by at least ϵ so that there is no gap left in the coverage. However, this implies that the left shift of S_j is at least $x + \epsilon$, contradicting the optimality of S'. Condition (c) is satisfied by S: Clearly if S' places S_i in position $p - \epsilon$ for some $\epsilon \geq 0$ then the left shift of S' is equal to $x + \epsilon$ contradicting the optimality of S'. If S' places S_i in position $p + \epsilon$ for some $\epsilon \geq 0$ then any placement of sensors $S_1, S_2, \ldots S_{i-1}$ in solution S' leaves a gap somewhere between 0 and p.
Condition (d) is satisfied: This is symmetric to Case 3.

Proof. (of Theorem 2) The optimality of the algorithm follows from Lemma 3. As far as the complexity of the algorithm is concerned, the algorithm needs to cover at most $n + 1$ gaps in $[0, L]$. In order to cover gap g_i, the algorithm must compute the $lsurplus(g_i)$, and adjust the shift values of sensors. This involves a scan of the shift values of the sensors that can be done in linear time. Thus, the time complexity of the algorithm is $O(n^2)$.

Proof. (of Theorem 3) Let the i-th gap be $[a_i, b_i]$. Let $lnode(g_i)$ be the index of the sensor node immediately to the left of gap g_i. Observe that $lnode(g_i) + 1$ is the sensor node immediately to the right of the gap g_i. For each gap g_i, we call $lsurplus(g_i)$ to be the *surplus sensor range* between the gap g_{i-1} and gap g_i to cover the gap g_i. Then $lsurplus(g_i) = (lnode(g_i) - lnode(g_{i-1})) * 2r - (b_i - a_i)$. On the other hand, $rsurplus(g_i)$ is defined to be the surplus on the right of the gap g_i to cover the interval $[b_i, L]$. That is, $rsurplus(g_i) = (n - lnode(g_i))2r - (L - b_i)$. Note the asymmetry in the definitions of $lsurplus$ and $rsurplus$.

We define a procedure $RightBlockMove(i, j)$ that puts sensor S_i at position j, then moves sensor S_{i-1} to $i - 2r$ and continues until finding a sensor that would

be forced to stay immobile or move left. A precondition for calling the procedure would be that such a sensor indeed exists. The procedure $LeftBlockMove(i, j)$ is defined analogously.

If $lsurplus(g_i) \geq 0$, using procedure $RightBlockMove(lnode(g_i), b_i - r)$, it follows that it is possible to cover the gap g_i with sensors entirely from the left of the gap g_i such that the maximum shift incurred by a sensor is $b_i - a_i$. In fact it is sensor $lnode(g_i)$ that will incur this shift; all other sensors will incur at most this shift. Similarly, if $rsurplus(g_i) \geq 0$, this means that it is possible to cover the interval $[b_i, L]$ using sensors only to the right of the gap g_i.

We now describe our algorithm for MinMax in a recursive manner. We show how to cover gap g_1 and then issue a recursive call to solve a smaller sub-problem.

1. If $lsurplus(g_1) \geq 0$ and $rsurplus(g_1) \geq 0$, then we can use procedure $RightBlockMove(lnode(g_1), b_1 - r)$ to cover the gap g_1 entirely with sensors from the left. Solve recursively the MinMax problem for the interval $[b_1, L]$ using sensors $lnode(g_1) + 1$ to n.
2. If $lsurplus(g_1) < 0$ and $rsurplus(g_1) \geq 0$, cover as much as possible of the gap with sensors from the left. More precisely, sensor node S_i should move to position $(2i - 1)r$. This can be achieved by using the procedure $RightBlockMove(lnode(g_1), j)$ to cover the gap up to position j where $j = (lnode(g_1))2r - r$. Next, use $LeftBlockMove(lnode(g_1) + 1, j)$ to cover the remaining gap using sensors originally on the right of gap g_1. Suppose the rightmost sensor to move left in this procedure was S_k, and the gap immediately after that was g_t. We now recursively solve the MinMax problem on the interval $[x_{k+1} - r, L]$ using sensors $k + 1$ to n. Note that $lsurplus(g_t)$ is recalculated to take into account only sensors to the right of S_k. In other words, we reassign $lnode(g_{t-1}) = k$.
3. Otherwise $lsurplus(g_1) > 0$ and $rsurplus(g_1) < 0$. In this case, some sensors from the left of gap g_1 will have to move to the right of the gap, since there aren't enough sensors on the right of the gap to cover the interval $[b_1, L]$. Let $k = \lceil rsurplus(g_1)/2r \rceil$ and let $j = (n - lnode(g_1) - k)2r$. We move sensors $lnode(g_1) - k + 1$ to $lnode(g_1)$ to the position j. This means the interval $[j, L]$ can be covered exactly by the sensors now in that range using the algorithm for $R = L$ given in this section. Finally we use $RightBlockMove(lnode(g_1) - k, b_1 - r)$ to cover the remaining part of g_1 entirely from the left.

We now argue that this algorithm is 2-optimal. Observe that the sensors that are moved to the right end of a gap in Step 3 are making *necessary moves*; since $rsurplus(g_i) < 0$, it is not possible to cover the interval to the right of g_i using only sensors whose initial positions were greater than b_i. Since the sub-problem will now we solved using the optimal algorithm for $R = L$ given in this section, it is easy to see that the total right shift incurred by these sensors is optimal. For all other sensors, observe that they are moved at most once during the algorithm. We now argue that every such move of a sensor during the algorithm causes a shift that is at most twice the maximum shift produced by the optimal algorithm.

First notice that in Step 1, while considering gap g_i, $RightBlockMove$ always incurs a shift of at most g_i, while $\max\{g_i/2\}$ is a lower bound on the maximal shift incurred by the optimal algorithm. Second, in Step 2, suppose the procedure call $LeftBlockMove(lnode(g_1)+1, j)$ results in covering the interval $[j, x_{k+1} - r]$ (recall that S_k was the rightmost sensor to move left in the course of the procedure) and results in a maximum left shift of s. We claim that $s/2$ must then be a lower bound for the maximum shift incurred by the optimal algorithm. This is because any decrease in the left shift experienced by sensors covering this interval must come at the expense of an equal increase in right shift of other sensors.

Finally in Step 3, consider the moves made by sensors in the set S' whose initial positions were in the interval $[b_1, L]$. Since the sub-problem is solved optimally, clearly an optimal algorithm for the entire problem could only have improved matters by using more sensors from the left of g_1 to cover the interval $[b_1, L]$. The maximum right shift of sensors in S' cannot be improved this way. Suppose the maximum left shift experienced by sensors in S' in our algorithm is s. Then any decrease in the left shift of sensors in S' in the optimal algorithm must come with a corresponding and equal increase in the right shift of other sensors. This implies that $s/2$ is a lower bound on the maximum shift.

Since the maximum shift incurred by a sensor in our algorithm is always at most twice the maximum shift incurred by sensors in the optimal algorithm, our algorithm is 2-optimal.

Clearly, there are at most $n+1$ gaps to be considered. Each sensor can be involved in the calculation of $lsurplus(g_i)$ only for one value of i. The value $rsurplus(g_1)$ is calulated at cost $O(n)$ and then it is adjusted for $rsurplus(g_i)$, $i > 1$ with a constant cost. Thus the algorithm is linear.

Proof. (of Theorem 5) Next, we describe how to find the set L in an inductive manner. Assume that the optimal solution for the sensors $\{S_1, \ldots, S_i\}$ is given by a set of intervals $\{L_1, L_2, \ldots, L_j\}$ where $j \leq i$. For each L_t, let $ms(L_t)$ be the value of the maximum shift incurred by a sensor in L_t. We now show how to extend the solution to include the sensor S_{i+1}.

If $I(S_{i+1}, x_{i+1})$ does not overlap with L_j, then clearly the sensor S_{i+1} should not move at all, that is, the optimal solution for the set $S_1, \ldots, S_{i+1}$ is the set of intervals $\{L_1, \ldots, L_j, L_{j+1}\}$ with $L_{j+1} = I(S_{i+1}, x_{i+1})$. On the other hand, if $I(S_{i+1}, x_{i+1})$ does overlap with L_j, we need to combine the two intervals L_j and $I(S_{i+1}, x_{i+1})$ to remove the overlap while keeping the maximum shift in the combined solution as small as possible.

We briefly describe how to combine two intervals P and Q assuming $le(P) \leq le(Q) \leq re(P) \leq re(Q)$. The combined interval will be assigned to P. We assume that $ms(P) \geq ms(Q)$. We maintain the invariant that the maximum left shift over all sensors is the same as the maximum right shift over all sensors processed so far. Let the overlap between the two intervals be $c = re(P) - le(Q)$. If $ms(P) - ms(Q) \geq c$, we push Q by c to the right, and attach it to the right end of P. Clearly in this case, the left shift of sensors originally in Q is not increased, and the right shift of such sensors is at most $ms(P)$. Since P is not moved, this does

not increase the value of $ms(P)$. If instead $ms(P) - ms(Q) < c$, we push P by $(c - ms(P) + ms(Q))/2$ to the left and Q by $(c + ms(P) - ms(Q))/2$ to the right. It is easy to verify that $ms(P) = (c + ms(P) + ms(Q))/2$, and is in fact the value of the maximum left shift of sensors originally in P as well as the maximum right shift of sensors originally in Q. The case when $ms(P) < ms(Q)$ is similar. The optimality of the combine procedure follows from the maintenance of the invariant and the fact that the two intervals are now exactly adjacent.

We use the above procedure to combine the intervals L_j and $I(S_{i+1}, x_{i+1})$. If the combined interval is disjoint from L_{j-1} we can stop, otherwise, we need to combine again with L_{j-1} repeating as long as necessary.

It remains to analyze the complexity of the algorithm. The combine procedure clearly takes $O(1)$ time, so it comes down to analyzing the number of times the combine procedure is called. We charge the operation $combine(P, Q)$ to the leftmost sensor in Q. Since after the two intervals are merged, that sensor is never again the leftmost sensor in an interval, it is clear that each sensor can be charged at most once. This means the complexity of the algorithm is linear in the number of sensors.

Finally, we outline the algorithm for the case when the final positions of the sensors are required to be within the interval $I = [0, L]$. First we use the algorithm detailed above to solve the problem on the infinite line. If the result falls entirely within I, we are done. Suppose instead that some of the intervals in the output set L lie to the left of the interval I. Then we start with L_1 and push it to the right, attaching it to any interval it encounters, pushing the attached interval, and continuing until $le(L_1) = 0$. If some of the intervals in L lie to the right of the interval I, we do a similar procedure from the rightmost interval in L. The resulting intervals must constitute a valid solution (no overlapping intervals), since $R < L$.